A Casual Analysis of Public Transportation in New York City

Author: Nathaniel del Rosario, A17562063

Rubric

Introduction, Question(s), & Hypothesis

The New York City public transportation is arguably one of the best in North America, providing many different methods such as metro, ride share, and bike as the most common. However, it is not a perfect system, possessing its own set of shortcomings. For example, compared to Tokyo's public transportation infrastructure, NYC's system is not as expansive and under serves more areas compared to Tokyo. Considering such context, the question arises, "just how under served are parts of New York City in the scope of public transportation?" and furthermore, are there any effects in other domains due to these under served areas?

I hypothesize that there are in fact different factors whose effects that are correlated with some areas being under served specifically by the NYC metro such as these areas being more likely to experience more ride share and bike usage. Upon witnessing any correlation, the next question becomes “is there causation as well?” Answering such uncertainty is the goal of this project.

This question is important because it involves using population, ridership, geo-spatial, and tract data to help people not only understand their commute as well as identify potential causality between different events and transportation accessibility. On average people will spend at least an hour commuting to and from work and school, and this is a huge chunk of our day (1/16 if you get a full 8 hours of sleep!) Additionally, public transportation companies can benefit greatly from this analysis as they can modify their strategy to appeal more to commuters and plan where to expand service to those who are under served. Lastly, the average citizen would benefit from this information because it could convince them to take public transportation instead of contributing to the increasing problem of traffic congestion in major metropolitan areas.

Related Work

Scarlett T. Jin, Hui Kong & Daniel Z. Sui (2019) Uber, Public Transit, and Urban Transportation Equity: A Case Study in New York City, The Professional Geographer, 71:2, 315-330, DOI: 10.1080/00330124.2018.1531038

distribution of Uber services is highly unequal, Correlation analysis shows that there tend to be fewer Uber pickups in low-income areas

Tang, J.; Gao, F.; Liu, F.; Zhang, W.; Qi, Y. Understanding Spatio-Temporal Characteristics of Urban Travel Demand Based on the Combination of GWR and GLM. Sustainability 2019, 11, 5525. https://doi.org/10.3390/su11195525 (https://doi.org/10.3390/su11195525)

results suggest that most taxi trips are concentrated in a fraction of the geographical area. Variables including road density, subway accessibility, Uber vehicle, point of interests (POIs), commercial area, taxi-related accident and commuting time have significant effects on travel demand,

Packages & Libraries

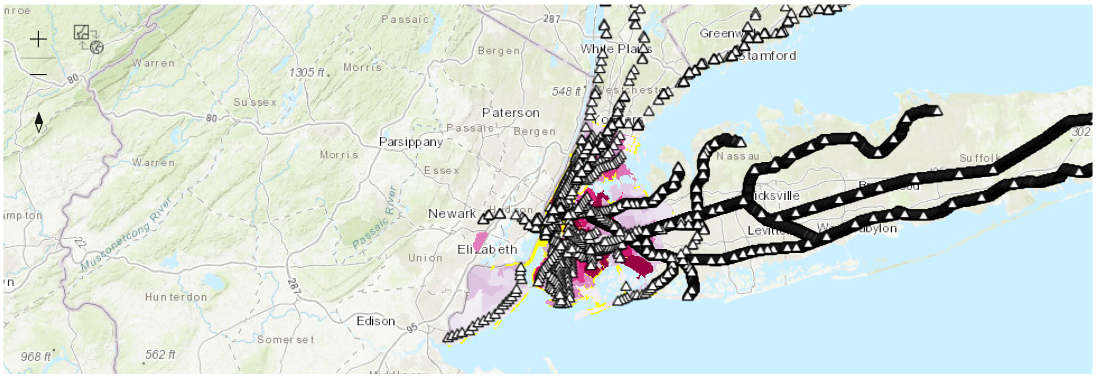
The packages and libraries used for our analysis are ArcGIS Online (including all ArcGIS features and analysis functions, Python, GeoPandas, shapely Point geometry)

Data Sources

In [1]:
```
import pandas as pd
import numpy as np
import math
import geopandas as gpd
import matplotlib.pyplot as plt
from shapely.geometry import Point

import arcgis
from arcgis.gis import GIS
from arcgis import geometry
from arcgis.features import GeoAccessor, GeoSeriesAccessor, FeatureLayerCollection, FeatureSet, FeatureCollection, FeatureLayer
from arcgis.features.use_proximity import create_buffers
from IPython.display import display
import os

gis = GIS("https://ucsdonline.maps.arcgis.com/home/index.html", "dsc170wi24_", "")
```

In [2]:
```
m = gis.map('New York, NY')
```

In [3]:
```python
# get transport feature layers
metro_stops_fl = gis.content.get('d52e004c3bda4397ae2145257ede1200')
rideshare_fl = gis.content.get('072e86100593482887a99aaaac8b2ada')
bike_lanes_fl = gis.content.get('8aff6fb97ef546679e97b1696bfbf052')
bike_lane_low_income_intersect_fl = gis.content.get('dc2e07a9af82464e94318c7dc71fc084')
bike_station_low_income_intersect_fl = gis.content.get('f0679d1e4ca44350abed2a48eecb7eb9')

# get income layers
low_income_binary_fl = gis.content.get('9bb695ac4b874286ab6645e4196f19bb')
income_dist_fl = gis.content.get('00847778292e466082388a18230f41ba')
gentrification_fl = gis.content.get('f8f47e4166d34862a6d340d8e2dcb55f')

m.add_layer(metro_stops_fl)
m.add_layer(rideshare_fl)
m.add_layer(bike_lanes_fl)
m
```



In [4]:
```python
ppend to each feature service: /0/query?where=1%3D1&outFields=*&f=geojson
ncome: https://services1.arcgis.com/HmwnYiJTBZ4UkySc/arcgis/rest/services/NYCMedianIncomeDistributions_WFL1/FeatureServer/0/query?where=1%3D1&
he rest are uploaded to https://github.com/natdosan/causal-analysis-nyc-transit
```

In [5]:
```python
# Load Data into GeoDataFames
bike_stations = gpd.read_file('data/bike_stations.json')
bike_lane_low_income_intersections = gpd.read_file('data/bike_lane_low_income_intersections.json')
low_income_bike_station_intersections = gpd.read_file('data/low_income_bike_station_intersections.json')
uber_lyft_dropoffs = gpd.read_file('data/uber_lyft_dropoffs.json')
nyc_subway_stops = gpd.read_file('data/nyc_stations.json')
low_income_census = gpd.read_file('data/low_income_census.json')
nyc_gentrification = gpd.read_file('data/nyc_gentrification.json')
nyc_median_income = gpd.read_file('data/nyc_median_income.json')
nyc_boundaries = gpd.read_file('data/nyc_boundaries.json')
```

In [6]:
```python
uber_lyft_dropoffs.head(1)
```

Out[6]:

	OBJECTID_1	OBJECTID	Shape_Leng	zone	LocationID	borough	count_	DOLocationID	time_day	PULocationID	Shape__Area	Shape__Length	geometry
0	1	1	0.116357	Newark Airport	1	EWR	257837.0	1.0	Morning	NaN	7.903953e+07	37646.072282	POLYGON ((-74.18445 40.69500, -74.18449 40.695...

Analysis

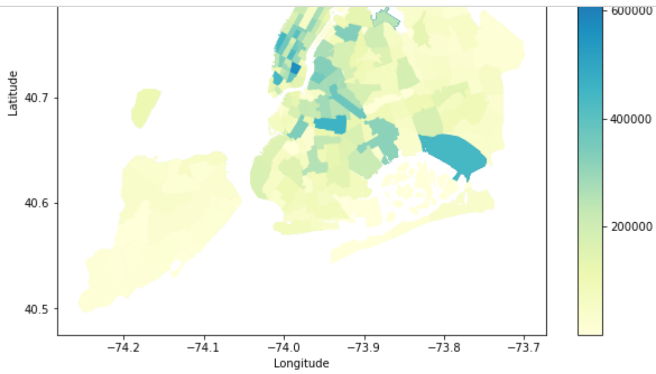
Outline

- Rideshare dropoffs by Tract
- Create Buffers for Bike and Metro Stations
- Overlay Buffers for each with Rideshare Dropoffs Choropleth
- Overlay Buffers for each with Income Choropleth
- Aggregate Buffers per Tract for Bike and Metro in a Choropleth

First we define public transportation as metro, bike, and uber. Walking and driving are non-public transporation. Keep this in mind as we go further with each analysis step

Rideshare Dropoffs by Tract

In [7]:
```
# Create Rideshare Choropleth
uber_lyft_dropoffs.plot(column='count_', cmap='YlGnBu', figsize=(10, 8), legend=True)
plt.title('Rideshare Dropoffs in NYC in 2023')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```



Now that we have a rideshare choropleth for dropoffs by tract, lets look at the metro stop locations / density, as well as bike lines / stations in relation to rideshare dropoffs

For both cases, I used the size (specifically length since most blocks are rectangles) of a manhattan block as the buffer radius. I did not divide by 2 because I set the criterion to be that each station is within 2 block lengths to be the buffer size.

The same was done for citi bike stations.

In [8]:
```
# Look at Metro, Bike, Rideshare Intersections -> Create Buffers for Metro Stations, Bike Statinos

# Calculate buffer distance in relation to Manhattan blocks (used google for a rough estimate)
latitude_nyc = 40.7128
longitude_nyc = -74.0060
block_width_meters = 264 * 0.3048
block_length_meters = 900 * 0.3048
latitude_radians = math.radians(latitude_nyc)
longitude_radians = math.radians(longitude_nyc)

# Calculate the conversion factors for latitude and longitude
# I asked ChatGPT how to convert from meters to latitude / longitude degrees
latitude_conversion_factor = 111132.92 - 559.82 * math.cos(2 * latitude_radians) + 1.175 * math.cos(4 * latitude_radians) - 0.0023 * math.cos
longitude_conversion_factor = 111412.84 * math.cos(latitude_radians) - 93.5 * math.cos(3 * latitude_radians)

# Convert block width and length to the same scale as latitude and longitude
block_width_degrees = block_width_meters / longitude_conversion_factor
block_length_degrees = block_length_meters / latitude_conversion_factor
```

In [9]:
```
for gdf in [bike_lane_low_income_intersections, low_income_bike_station_intersections, uber_lyft_dropoffs, nyc_subway_stops, low_income_censu
    print(f'CRS: {gdf.crs}')
```

```
CRS: epsg:4326
CRS: epsg:4326
CRS: epsg:4326
CRS: epsg:4326
CRS: epsg:4326
CRS: epsg:4326
CRS: epsg:4326
```

In [10]:
```
nyc_subway_stops.head(1)
```

Out[10]:

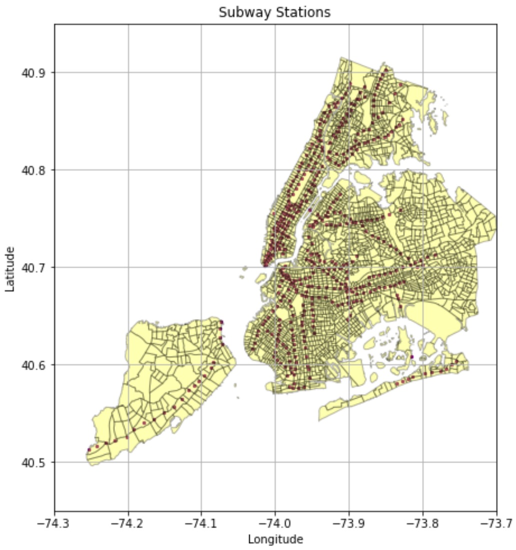
	FID	stop_id	stop_name	stop_lat	stop_lon	trains	structure	stop_id2	GEOID	NAMELSAD	geometry
0	1	101	Van Cortlandt Park - 242 St	40.889248	-73.898583	1	Elevated		36005	Bronx County	POINT (-73.89858 40.88926)

In [11]:
```
bike_stations.head(1)
```

Out[11]:

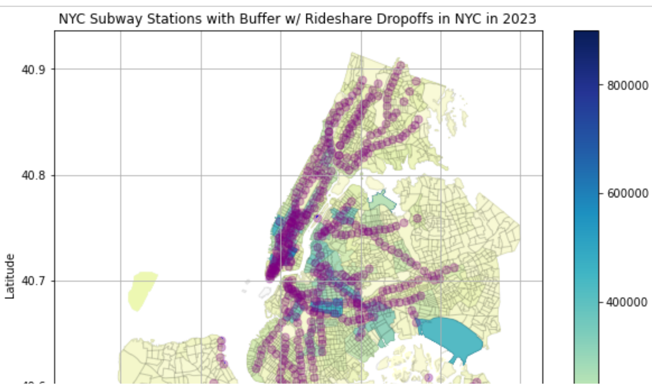
	OBJECTID	tripduration	starttime	stoptime	start_station_id	start_station_name	start_station_latitude	start_station_longitude	end_station_id	end_station_name	end_station_latitude	end_stati
0	1	330	00:01.5	05:31.8	3602	31 Ave & 34 St	40.763154	-73.920827	3570	35 Ave & 37 St	40.755733	

In [12]:
```python
# Plot the subway stations
nyc_subway_stops.plot(marker='o', color='purple', markersize=5, figsize=(10, 8))
plt.title('Subway Stations')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid(True)
nyc_boundaries.plot(ax=plt.gca(), alpha=0.3, color='yellow', edgecolor='black')
plt.xlim(-74.3, -73.7)
plt.ylim(40.45, 40.95)
plt.show()
```



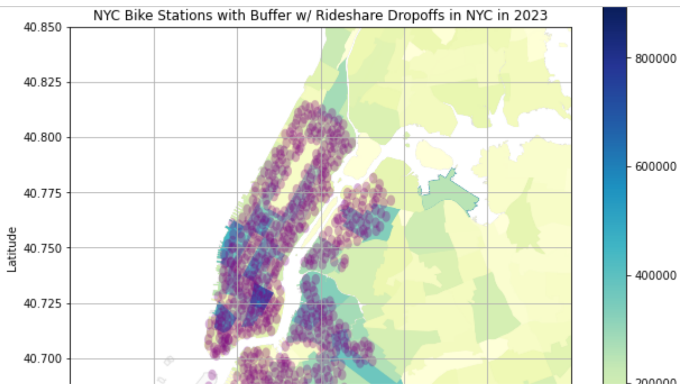
Create Buffers for Metro Stations Overlayed On Rideshare Dropoffs

In [13]:
```python
# Buffer around NYC subway stations
nyc_subway_buffer = nyc_subway_stops.buffer(block_length_degrees)
nyc_subway_stops['buffer'] = nyc_subway_buffer

# create map object: plot metro stop buffers
map1 = nyc_subway_stops['buffer'].plot(color='blue', alpha=1, figsize=(10, 8))
uber_lyft_dropoffs.plot(ax=map1, column='count_', cmap='YlGnBu', figsize=(10, 8), legend=True)
nyc_boundaries.plot(ax=map1, color='lightgray', alpha = .1, edgecolor='black')
nyc_subway_stops.plot(ax=map1, marker='o', color='purple', markersize=50, alpha = .3)

plt.title('NYC Subway Stations with Buffer w/ Rideshare Dropoffs in NYC in 2023')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid(True)
plt.show()
```



Create Buffers for Bike Stations Overlayed On Rideshare Dropoffs

In [14]:
```python
# Buffer around bike stations
bike_stations_buffer = bike_stations.buffer(block_length_degrees)
bike_stations['buffer'] = bike_stations_buffer

# create map object: plot bike statino buffers
map2 = nyc_boundaries.plot(color='lightgray', alpha = .1, edgecolor='black', figsize=(10, 8))
uber_lyft_dropoffs.plot(ax=map2, column='count_', cmap='YlGnBu', legend=True)
bike_stations['buffer'].plot(ax=map2, color='purple', alpha=0.3)
#bike_stations.plot(ax=map2, marker='o', color='purple', markersize=50, alpha = .3)


plt.title('NYC Bike Stations with Buffer w/ Rideshare Dropoffs in NYC in 2023')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.xlim(-74.1, -73.8)
plt.ylim(40.65, 40.85)
plt.grid(True)
plt.show()
```



As we can see with metro station / bike station buffers, a big portion of Queens is underserved in both regards. Brooklyn has a lot of bike stations as well as metro stops, and unsurprisingly Manhattan, especially lower Manhattan has the highest density of both bike stations and metro stops, with the buffers showing that nost of this region is covered with the 2 block radius criterion.

The next thing we will do is create choropleths for the metro station and bike station density. This will be useful later for our metric design.

Aggregating Buffers Per Census Tract and Per Income Tract

Motivation:

We aggregate buffers instead of stations because buffers allow for a general area to be covered. Specifically, with a station on the border of a tract, it will only be counted in the tract it is in. However, with buffers, any buffers on the edge of a tract within 2 manhattan grid lengths will be counted in both tracts. This promotes the idea of walkability and accessability, since it is unrealistic to have a station right outside your front door!

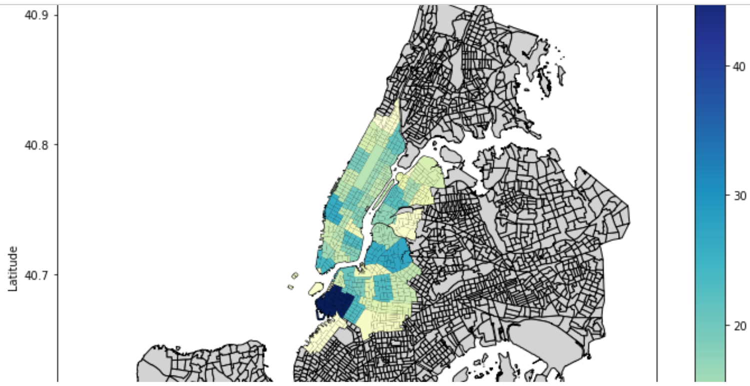
In [15]:
```python
nyc_boundaries.head(1) # our column to agg by is NTA2020
```

Out[15]:

	OBJECTID	CTLabel	BoroCode	BoroName	CT2020	BoroCT2020	CDEligibil	NTAName	NTA2020	CDTA2020	CDTANAME	GEOID	PUMA	Shape__Area	Shape__Length	geometr
0	1	1	1	Manhattan	000100	1000100	None	The Battery-Governors Island-Ellis Island-Libe...	MN0191	MN01	MN01 Financial District-Tribeca (CD 1 Equivalent)	36061000100	4121	1.842974e+06	10832.877284	MULTIPOLYGO (((-74.0438 40.6902 -74.04351

To start, let's look at just stations aggregated first, not their buffers:

In [38]:
```python
# 1. Perform spatial join
bike_stations_with_tracts = gpd.sjoin(gpd.GeoDataFrame(bike_stations), nyc_boundaries, how='inner', predicate='intersects')

# 2. Aggregate by census tract
bike_station_counts = bike_stations_with_tracts.groupby('NTA2020').size().reset_index(name='bike_station_count')

# 3. Merge aggregated counts back into nyc_boundaries
nyc_boundaries_with_counts = pd.merge(nyc_boundaries, bike_station_counts, on='NTA2020', how='left')

fig, ax = plt.subplots(figsize=(12, 10))
nyc_boundaries.plot(ax=ax, color='lightgray', edgecolor='black')
nyc_boundaries_with_counts.plot(ax=ax, column='bike_station_count', cmap='YlGnBu', figsize=(10, 8), legend=True)
plt.title('Bike Stations per Census Tract in NYC')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```



Wow, it looks like there are a lot underserved areas outside of downtown Brooklyn and Manhattan! Let's do the same for metro stations

In [39]:
```python
# 1. Perform spatial join
subway_stations_with_tracts = gpd.sjoin(gpd.GeoDataFrame(nyc_subway_stops), nyc_boundaries, how='inner', predicate='intersects')

# 2. Aggregate by census tract
subway_station_counts = subway_stations_with_tracts.groupby('NTA2020').size().reset_index(name='subway_station_count')

# 3. Merge aggregated counts back into nyc_boundaries
nyc_boundaries_with_subway_counts = pd.merge(nyc_boundaries, subway_station_counts, on='NTA2020', how='left')

fig, ax = plt.subplots(figsize=(10, 8))
nyc_boundaries.plot(ax=ax, color='lightgray', edgecolor='black')
nyc_boundaries_with_subway_counts.plot(ax=ax, column='subway_station_count', cmap='YlGnBu', figsize=(10, 8), legend=True)
plt.title('Subway Stations per Census Tract in NYC')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

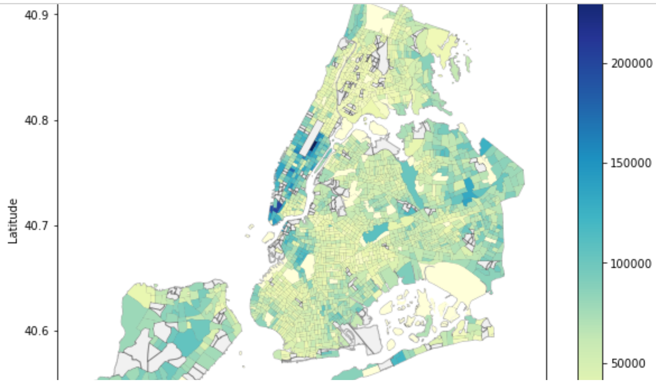
The first half of our analysis focused on the relationship between each method of transportation (Metro, Bike, and Rideshare) and the tracts / regions of NYC. Now we will look into whether income also correlates with these 3 methods of transportation.

Overlay Buffers for each with Income Choropleth

In [18]:
```python
# Look at income vs these intersections, (income vs metro, income vs bike, income vs rideshare)
# Do there apppear to be any correlations
```

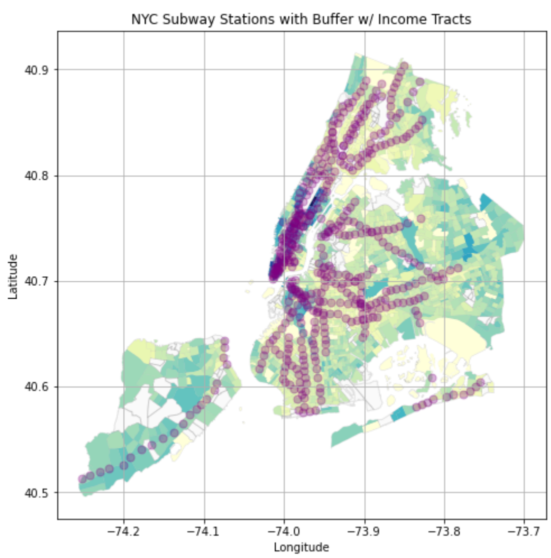
To begin this half, let's look at income by tract before doing any overlays

In [19]:
```
fig, ax = plt.subplots(figsize=(10, 8))
nyc_boundaries.plot(ax=ax, color='lightgray', edgecolor='black', alpha=.3)
nyc_median_income.plot(ax=ax, column='nyct2010_MedianInc', cmap='YlGnBu', legend=True)
plt.title('NYC Median Income By Tract')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

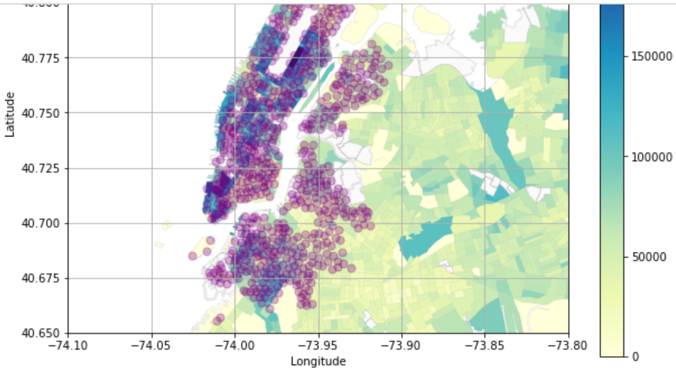


Next, we look at metro and bike station buffers overlaid on income tracts

In [27]:
```
# create map object: plot metro stop buffers on income tracts
fig, ax = plt.subplots(figsize=(10, 8))
nyc_boundaries.plot(ax=ax, color='lightgray', alpha = .1, edgecolor='black')
nyc_median_income.plot(ax=ax, column='nyct2010_MedianInc', cmap='YlGnBu')
nyc_subway_stops.plot(ax=ax, marker='o', color='purple', markersize=50, alpha = .3, legend=True)

plt.title('NYC Subway Stations with Buffer w/ Income Tracts')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.grid(True)
plt.show()
```



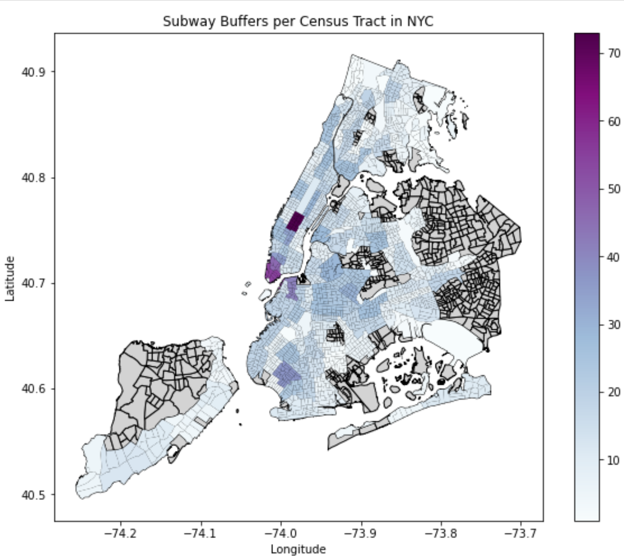
```
In [30]: # create map object: plot bike station buffers on income tracts
fig, ax = plt.subplots(figsize=(10, 8))
nyc_boundaries.plot(ax=ax, color='lightgray', alpha = .1, edgecolor='black')
nyc_median_income.plot(ax=ax, column='nyct2010_MedianInc', cmap='YlGnBu', legend=True)
bike_stations.plot(ax=ax, marker='o', color='purple', markersize=50, alpha = .3, legend=True)

plt.title('NYC Bike Stations with Buffer w/ Income Tracts')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.xlim(-74.1, -73.8)
plt.ylim(40.65, 40.85)
plt.grid(True)
plt.show()
```



Putting these plots together: Buffers Per Tract

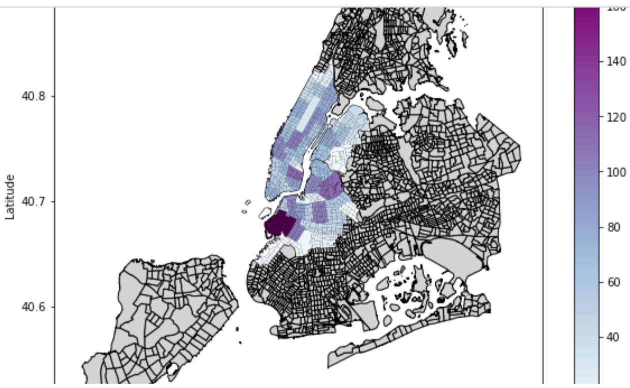
Since we can't really overlay the choropleth layers without using a library like plotly, we will plot the aggregated layers below:

```
In [37]: . Perform spatial join
vay_stations_with_tracts = gpd.sjoin(gpd.GeoDataFrame(geometry = nyc_subway_stops['buffer']), nyc_boundaries, how='inner', predicate='intersec

. Aggregate by census tract
vay_station_counts = subway_stations_with_tracts.groupby('NTA2020').size().reset_index(name='subway_station_count')

. Merge aggregated counts back into nyc_boundaries
_boundaries_with_subway_counts = pd.merge(nyc_boundaries, subway_station_counts, on='NTA2020', how='left')

, ax = plt.subplots(figsize=(10, 8))
_boundaries.plot(ax=ax, color='lightgray', edgecolor='black')
_boundaries_with_subway_counts.plot(ax=ax, column='subway_station_count', cmap='BuPu', figsize=(10, 8), legend=True)
.title('Subway Buffers per Census Tract in NYC')
.xlabel('Longitude')
.ylabel('Latitude')
.show()
```



In [36]:

```
# 1. Perform spatial join
subway_stations_with_tracts = gpd.sjoin(gpd.GeoDataFrame(geometry = bike_stations['buffer']), nyc_boundaries, how='inner', predicate='interse

# 2. Aggregate by census tract
subway_station_counts = subway_stations_with_tracts.groupby('NTA2020').size().reset_index(name='subway_station_count')

# 3. Merge aggregated counts back into nyc_boundaries
nyc_boundaries_with_subway_counts = pd.merge(nyc_boundaries, subway_station_counts, on='NTA2020', how='left')

fig, ax = plt.subplots(figsize=(10, 8))
nyc_boundaries.plot(ax=ax, color='lightgray', edgecolor='black')
nyc_boundaries_with_subway_counts.plot(ax=ax, column='subway_station_count', cmap='BuPu', figsize=(10, 8), legend=True)
plt.title('Subway Buffers per Census Tract in NYC')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```



As we can see from the legends, the number of buffers per tract is larger than the number of stations, which supports our initial hypothesis about promoting walkability

With EDA finally done, we can start combining layers!

In [23]:

```
# Design Score / Metric for each Greater NYC adminstrative boundary
# Accessability = weighted sum(.5 * metro, .25 * bike, .25 * rideshare)
# Compare with Income tracts to see if Low accessibility correlated w/ lower income
# (if yes: reasons could be farther distance from metro, bike, uber buffers, less development in these areas)
```

In [24]:

```
# Finally, Choropleth of Score by Administrative Boundary
```

In [25]:

```
# If time: regression to predict score based on income
```

Summary & Results

Discussion

Conclusions & Future Work